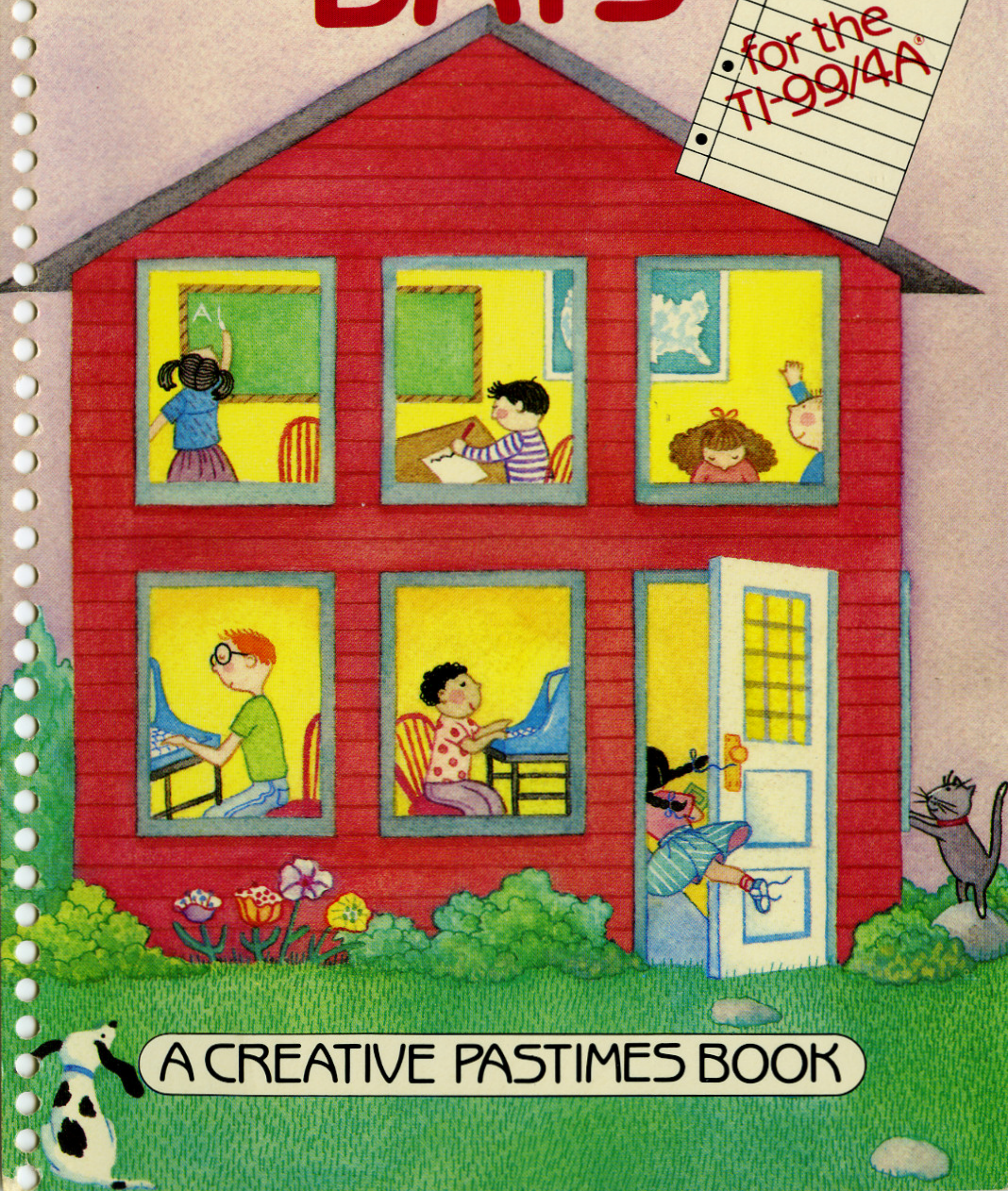
CLAIRE BAILEY PASSANTINO

SCHOOL DAYS

for the TI-99/4A®



A CREATIVE PASTIMES BOOK

School Days
for the TI-99/4A® Computer

Claire Bailey Passantino

Text Illustrations by Nancy Gurganus

A Reston Computer Group Book
Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia

A CREATIVE PASTIMES BOOK

Library of Congress Cataloging in Publication Data

Passantino, Claire Bailey.
School days for the TI-99/4A computer.

(A Creative pastimes book)
Summary: Presents twenty programs that include games, contests, riddles, songs, and other activities to use with BASIC on the TI-99/4A computer.
1. Computer games—Juvenile literature. 2. TI-99/4A (Computer)—Programming—Juvenile literature. 3. Basic (Computer program language)—Juvenile literature. [1. Computer games. 2. TI-99/4A (Computer)—Programming. 3. Basic (Computer program language) 4. Programming (Computers)] I. Gurganus, Nancy, ill. II. Title. III. Series.
GV1469.2.P373 1984 794.8′2 84-3470
ISBN 0-8359-6887-1

Cover illustration by Bethann Thornburgh
Cover design by Carol Conway

10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

Other Itty Bitty Bytes Books:

Itty Bitty Bytes of Space
Matilda, the Computer Cat

These titles are also available for the ATARI® Computer and the Commodore 64® Computer.

Contents

Foreword

No matter how much kids hate school, there are some things about it that are undeniably fun. Hopefully the programs in this book will make you think fondly of your own days at school.

School Days is the third of the Itty Bitty Bytes books for the TI-99/4A® Computer. Each book in the series is designed to bring you twenty fun programs and, at the same time, teach you something about BASIC. Typing in the programs is hard work. You must be careful to number the lines exactly, spell the words correctly, and put in all the right punctuation. But you will feel extra good when your program runs smoothly!

Always read the explanations next to each program. Sometimes there are additions or changes that you can make to improve the program. Once you see what the program is all about, feel free to experiment with it—and by all means combine programs that work well together. The more you can do to make a program reflect your own special personality, the more you will enjoy it. Let me know if you find some nifty changes!

The Itty Bitty Bytes books have grown out of the teaching experiences I've had with my computer students. Special thank-yous go to all my COMPU-KIDS—and to my own kids, too—who keep bringing me one good idea after another. They are my severest critics, but my most outstanding asset.

Happy computing!

Claire Bailey Passantino

TI-99/4A® is a registered trademark of Texas Instruments.

A Note to Parents and Teachers

You bought the computer. You read the manuals. You did the demo programs. You and/or your children may even have taken some computer classes. But now the computer is just sitting there. Everyone was so enthusiastic. *What happened?*

The novelty of a new computer will take you just so far. Beyond that point, a continuing interest in this incredible tool is directly related to its usefulness. "Not useful" equates to "not used." So the problem becomes, what can computers do that children would find useful? What kinds of things would encourage children to expend the energy needed to create their own computer programs?

Each Itty Bitty Bytes book in the Creative Pastimes series is packed with computer activities appealing to young programmers. Simple games, contests, races, pictures, designs, songs, riddles, charts, tests, and more—all are designed to be fun while reinforcing beginning computer concepts and skills.

Besides providing fun, there are fringe benefits to having children write their own programs. In working with children, I have found that computer programming encourages them to:

- Think creatively.
- Use logical thinking skills.
- Attend to details.
- Take small steps to achieve a goal.
- Personalize programs.
- Develop pride and self-esteem.
- Appreciate packaged software.

To help children enjoy doing their own programs, here are some suggested DOs and DON'Ts:

- DO encourage children to type in their own programs. With younger children, bargain: "You type this line and I'll type the next one." (Save the long lines for yourself!)
- DON'T criticize typing expertise. Speed and correct fingering are typing skills that are minimally related to computer programming.
- DO allow children to make mistakes.
- DO help them find the errors they've made. (This is called "debugging" the program.)
- DON'T worry when there is an error message. This means that a mistake has been made. Check the program for "bugs."
- DO encourage children to read and understand the program explanations.

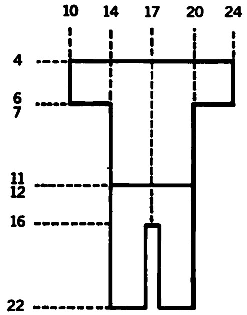
- DON'T, however, force the issue. Some people learn by reading. Others learn by doing. As skills are repeated over and over in different contexts, children may just "catch on."
- DO be aware of some common pitfalls. Remember to:
  1. Use line numbers.
  2. Press ENTER after you type in a line.
  3. Give great attention to spelling and punctuation. Quotation marks, commas, semicolons, colons, and even spaces are very important.
  4. Save your program before you turn off the computer.
- DO help children save their programs on tape or disk so they can use them again and again. If you have a printer, use it to make "hard" copies of each program. People like to see themselves in print.
- DON'T type the "FCTN" and the plus (+) keys simultaneously. This combination will erase the program from memory and you will have to start over.
- DO remember that typing the "FCTN" and the "4" simultaneously will break the program and get you out of an infinite loop.
- DO type in and run all programs with the ALPHA LOCK on (that is, in the *down* position). The only exceptions are the joystick programs, which are run with the ALPHA LOCK off.
- DO praise children for a job well done. And enjoy the programs that they've created.
- DO modify and use programs that you yourself find useful.

It is my sincere hope that the Itty Bitty Bytes books will help you and your children establish a healthy working relationship with your computer. Take that computer off the shelf! And let me know how things are working out!

School Days

for the TI-99/4A® Computer

Main ideas: Simple graphics
Using CALL KEY for instant input from the keyboard



This is a diagram of the outfit and the key coordinates used to create it. It may also be helpful for you to refer to the Character Sets and the Graphics Sheet in the Appendix. In this program, character 66 (B) from set 5 is used to draw the bottom of the outfit. Character 84 (T) from set 7 is used to draw the top of the outfit. Character 76 (L) from set 6 is used to make the legs by drawing a line down the middle of the pants. The program is divided into the following 4 sections.

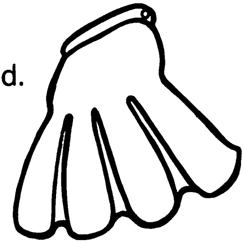
10 A remark that this section will set things up.
20–30 Sets the random numbers and clears the screen.
40 Makes the screen white (color 16). You may pick a different color.
50 The slit down the pants will be transparent (color 1). By choosing transparent, the slit will come out no matter what color you choose for the screen.
60 Sets the color of set 5 to light blue (color 6).
70 Sets the color of set 7 to light red (color 10).

80 A remark.
90–110 Makes 3 rows of character 84, beginning at column 10 and extending for 15 boxes (through box 24).
120–140 Makes 5 rows of character 84, beginning at column 14 and extending for 7 boxes (through box 20).

150 A remark.
160–180 Makes 11 rows of character 66, beginning at column 14 and extending for 7 boxes (through box 20).
190 Makes a vertical column beginning at row 16, column 17 and extending downward for 7 boxes. The line is made with character 76, which has been made transparent (in line 50), with all the other characters in set 6.

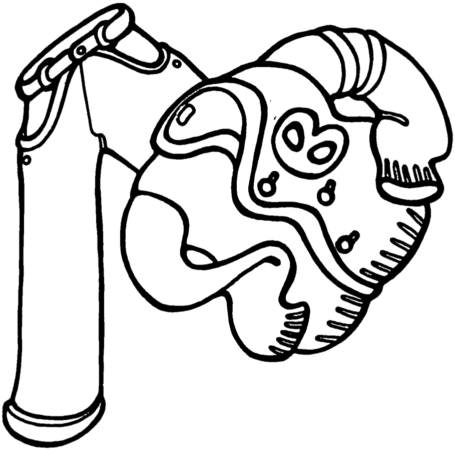
200 A remark.
210 Sets up the keyboard for instant input.
220 If you haven't pressed any key, sends you back to line 210 until you do press a key.
230–240 Picks a color from 1 to 16 for the top (T) and the bottom (B).
250–260 Changes the colors of sets 5 and 7 according to the colors picked in lines 230 and 240.
270 Goes back to line 210 to wait for you to press another key. Press FCTN–4 to stop the program.

Clothing the Bod



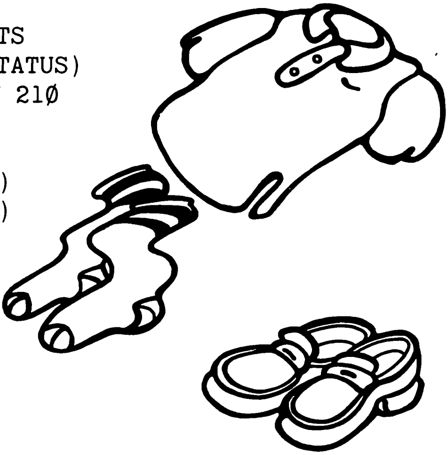
It's time for school. You can't go naked. Better find something to wear!

```
1Ø REM SET THINGS UP
2Ø RANDOMIZE
3Ø CALL CLEAR
4Ø CALL SCREEN(16)
5Ø CALL COLOR(6,1,1)
6Ø CALL COLOR(5,6,6)
7Ø CALL COLOR(7,1Ø,1Ø)

8Ø REM DRAW TOP
9Ø FOR R=4 TO 6
1ØØ CALL HCHAR(R,1Ø,84,15)
11Ø NEXT R
12Ø FOR R=7 TO 11
13Ø CALL HCHAR(R,14,84,7)
14Ø NEXT R

15Ø REM DRAW BOTTOM
16Ø FOR R=12 TO 22
17Ø CALL HCHAR(R,14,66,7)
18Ø NEXT R
19Ø CALL VCHAR(16,17,76,7)
```



When you add the last section, you may press any key to change the color of your outfit.

```
2ØØ REM CHANGE OUTFITS
21Ø CALL KEY(Ø,KEY,STATUS)
22Ø IF STATUS=Ø THEN 21Ø
23Ø T=INT(16*RND)+1
24Ø B=INT(16*RND)+1
25Ø CALL COLOR(5,B,B)
26Ø CALL COLOR(7,T,T)
27Ø GOTO 21Ø
```



Main ideas: Writing a computer song

200 Clears the screen.

205 Makes the screen white (color 16).

210 Makes the characters in set 2 red (color 9).

211 Makes the characters in set 3 blue (color 5).

212 Makes the characters in set 4 black (color 2).

220–240 Makes 7 red stripes at every other row beginning at row 3 and ending at row 15. Each row begins at column 6 and extends for 20 boxes.

250–270 Draws a solid blue rectangle by making 7 blue rows beginning at row 3 and ending at row 9. Each row begins at column 6 but extends for only 10 boxes, halfway across the flag.

280 Makes a vertical line for a flagpole.

290 Skips 9 spaces and prints the word AMERICA.

300 Reads the note (N) and the time (T) or number of beats to play the note.

301 If the number "−1" is read, you skip to line 330.

310 Plays the note, giving each beat 200 milliseconds. To speed up the song, choose a lower number. To slow it down, choose a higher number.

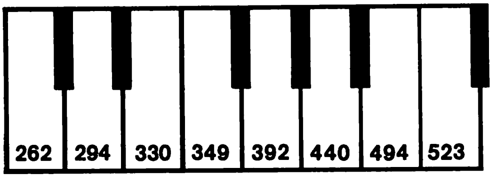
320 Goes back to line 300 and reads more data.

330 Ends the program.

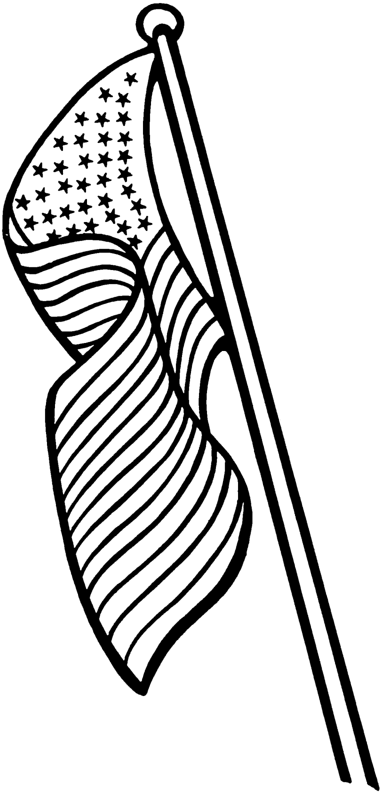
350–380 The data lines that hold the information for the song. Each line holds a series of numbers arranged in pairs—note, time, note, time, note, time, etc.

390 The flag data referred to in line 301.

This diagram holds all the notes used in the song:

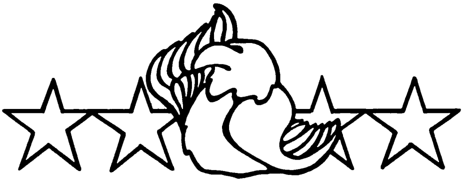


The American Way



School days in America all begin the same way. Everyone says the pledge to the flag. Then comes a patriotic song. Good morning, boys and girls!

```
200 CALL CLEAR
205 CALL SCREEN(16)
210 CALL COLOR(2,9,9)
211 CALL COLOR(3,5,5)
212 CALL COLOR(4,2,2)
220 FOR R=3 TO 15 STEP 2
230 CALL HCHAR(R,6,42,20)
240 NEXT R
250 FOR R=3 TO 9
260 CALL HCHAR(R,6,50,10)
270 NEXT R
280 CALL VCHAR(3,6,58,22)
290 PRINT "         AMERICA"
300 READ N,T
301 IF N=-1 THEN 330
310 CALL SOUND(200*T,N,2)
320 GOTO 300
330 END
350 DATA 262,2,262,2,294,2,247,3,262,1,294,2
355 DATA 330,2,330,2,349,2,330,3,294,1,262,2
360 DATA 294,2,262,2,247,2,262,6
365 DATA 392,2,392,2,392,2,392,3,349,1,330,2
370 DATA 349,2,349,2,349,2,349,3,330,1,294,2
375 DATA 330,2,349,1,330,1,294,1,262,1,330,3,349,1,
    392,2
380 DATA 440,1,349,1,330,2,294,2,262,6
390 DATA -1,-1
```



Main ideas: Inputting variables into a text
Using punctuation and spaces in a printout
Using FOR...NEXT to repeat an action

In this program, the following variables are used. Notice that all the alpha variables (the variables containing letters) have a dollar sign ($). They are called *string variables.* N is a *numeric variable* and has no dollar sign.

TN$ = Teacher's name
N$ = Your name (name of student)
S$ = School subject
RN$ = Relative's name
A$ = Animal
N = Number
PL$ = Place
PN$ = Parent's name

10 Clears the screen.
20–81 Allows you to input the information needed. Notice that the semicolon (;) makes the input appear right after the prompt. There is no semicolon at the end of line 40. Did you notice the difference?

100 Clears the screen.
110–210 Prints the letter. Be careful about quotation marks, semicolons, and spaces if you want the letter to turn out right.

230–231 Pauses while the computer counts to 3000. This gives you time to read the letter.

Guess what happened when the teacher found out the truth!

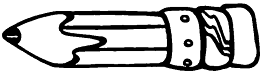
```
250 PRINT::
260 PRINT "PRESS H TO SEE WHAT ";N$
270 PRINT "HAD TO DO FOR HOMEWORK."
280 CALL KEY(0,K,ST)
290 IF ST=0 THEN 280
300 IF K<>72 THEN 280
310 FOR S=1 TO 500
320 PRINT S;"I WILL NOT TELL LIES."
330 NEXT S
```

How does this addition to the program work? First, the computer skips 2 lines and prints the directions. Line 280 sets up the keyboard for instant input. As soon as you press H (character 72), you go to line 310. The FOR...NEXT loop (lines 310 to 330) prints the sentence 500 times.

The Dog Ate It!

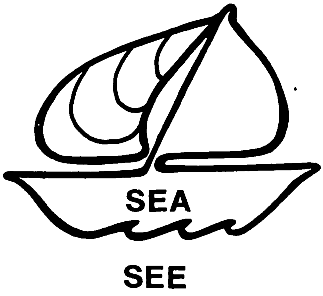
Oh no! You forgot to do your homework. It's a good thing you're so clever. Here's your homework excuse. You get an A for creative writing.

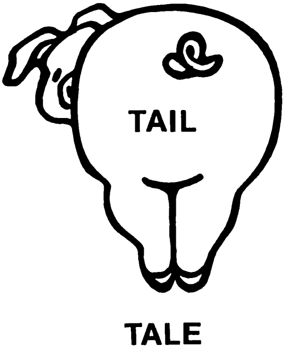
```
1Ø CALL CLEAR
2Ø PRINT "TEACHER'S NAME";
21 INPUT TN$
3Ø PRINT "YOUR NAME";
31 INPUT N$
35 PRINT "A SCHOOL SUBJECT";
36 INPUT S$
4Ø PRINT "NAME OF A RELATIVE"
41 INPUT RN$
5Ø PRINT "KIND OF ANIMAL";
51 INPUT A$
6Ø PRINT "A NUMBER";
61 INPUT N
7Ø PRINT "A PLACE";
71 INPUT PL$
8Ø PRINT "YOUR PARENT'S NAME";
81 INPUT PN$

1ØØ CALL CLEAR
11Ø PRINT "DEAR ";TN$;","
12Ø PRINT N$;" DID NOT DO THE "
125 PRINT S$;" HOMEWORK"
13Ø PRINT "LAST NIGHT BECAUSE"
14Ø PRINT RN$;" CAME TO VISIT."
15Ø PRINT
16Ø PRINT RN$;"'S ";A$
165 PRINT "GRABBED IT AND TOOK"
17Ø PRINT N;" BITES OUT OF IT."
18Ø PRINT "THEN HE RAN AWAY WITH IT TO "
19Ø PRINT PL$;"."
2ØØ PRINT TAB(12);"YOURS TRULY,"
21Ø PRINT TAB(12);PN$

23Ø FOR P=1 TO 3ØØØ
231 NEXT P
```



Main ideas: A test of homonyms
A test with two choices for each question

10 Clears the screen.
20 Prints the message.
30 Skips a line.
40 Reads 3 pieces of information: H$ (the hint for the homonym), CH$ (the choices), and A$ (the answer).
50 Checks for the flag data. (If it reads the word END in the H$ position, it skips to line 150.)
60 Prints the hint.
70 Prints the choices. (To make it harder, don't print the choices and skip this line entirely.)
80 Allows you to input a guess (G$).
90 Sends you to line 120 if you guessed correctly.
100 Tells you the right answer if you guessed incorrectly.
110 Skips over line 120 to line 130.
120 Prints "RIGHT!"
130 Skips a couple of lines.
140 Goes back to line 40 for the next homonym.
150–170 Ends the test.
180 The official end.
500–540 The data lines for the test. To make the test longer, add more data lines at any line numbers up to line 899. Be sure to follow the same pattern used in lines 500 to 540.
900 The flag data (referred to in line 50) that signals the program to end. The word END is written 3 times because line 40 must read 3 pieces of information.





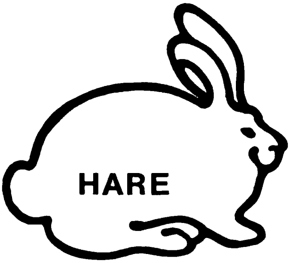
Spelling Counts!

Today's spelling lesson is about homonyms. Eye no watt a homonym is, dew ewe?

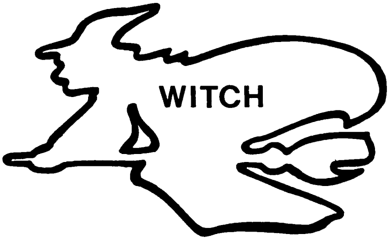
```
1Ø CALL CLEAR
2Ø PRINT "SPELLING COUNTS!"
3Ø PRINT
4Ø READ H$,CH$,A$
5Ø IF H$="END" THEN 15Ø
6Ø PRINT H$
7Ø PRINT CH$
8Ø INPUT G$
9Ø IF G$=A$ THEN 12Ø
1ØØ PRINT "WRONG, IT WAS ";A$
11Ø GOTO 13Ø
12Ø PRINT "RIGHT!"
13Ø PRINT::
14Ø GOTO 4Ø
15Ø PRINT
16Ø PRINT "THIS IS THE END."
17Ø PRINT "HOW DID EWE DEW?"
18Ø END
5ØØ DATA HALLOWEEN QUEEN,(WHICH/WITCH),WITCH
51Ø DATA EYES DO THIS,(SEE/SEA),SEE
52Ø DATA A RABBIT,(HARE/HAIR),HARE
53Ø DATA A GOOD STORY,(TAIL/TALE),TALE
54Ø DATA PENCILS DO THIS,(RIGHT/WRITE),WRITE
9ØØ DATA END,END,END
```

YOU

TO, TOO



HAIR



WHICH

Main ideas: Drilling arithmetic facts
Using the correct operators (+, −, *, /, =, <,>)

5–10 Randomizes the numbers and clears the screen.
20–30 Picks a number for X (1 to 10) and a number for Y (1 to 10).
40 Counts over 10 spaces and prints the number for X, a plus (+) sign, the number for Y, and an equal (=) sign. Then it advances the problem up the screen. (Experiment with the number of colons.)
50 Allows you to input a guess (G).
60 Sends you to line 200 if your guess was correct.
70 Prints this message if your guess was incorrect.
80 Goes back to line 50 so you can guess again.
200 This REM saves the line number for later use.
210 Counts over 20 spaces and prints.
220–230 Pauses while the computer counts to 500.
260 Keeps giving you new problems. Press FCTN–4 to stop.

Flashing your flash cards:

200 A remark to remind you that this is the flashing section.
210 Counts over 20 spaces and prints.
220–250 Makes 20 flashes:
230 Picks a number from 0 to 13, and adds 3 to that number. The result is a number from 3 to 16. This eliminates color 1 (transparent) and color 2 (black), which would make it impossible for you to read the black letters.
239–240 Makes the screen black, and then the color chosen in line 230.
250 Goes back to line 220 for the next flash.

Multiplication:

Note that the X is used to print the problem on the screen, but a star (*) is used when you want the computer to multiply.

Subtraction:

Line 35 is needed so that your answers are more than or equal to 0. For example, if the computer let X=3 and Y=5, then X−Y=−2. Line 35 eliminates the possibility of negative answers.

Division:

20 Picks a number from 0 to 100.
30 The +1 eliminates 0, since division by 0 is impossible.
35 Makes the dividend larger than the divisor.
36 Makes sure the answers are integers. For example, 13 ÷ 2 would be rejected because the answer is not a whole number.
40 Prints the problem.
60 Sends you to line 200 if you gave the right answer.

Flashy Flashcards

Mrs. Tiddlebaum's class loves math.

```
5 RANDOMIZE
10 CALL CLEAR
20 X=INT(11*RND)
30 Y=INT(11*RND)
40 PRINT TAB(10);X;"+";Y;"=":::::::::
50 INPUT G
60 IF G=X+Y THEN 200
70 PRINT "WRONG! TRY AGAIN!"
80 GOTO 50
200 REM
210 PRINT TAB(20);"RIGHT!";
220 FOR P=1 TO 500
230 NEXT P
260 GOTO 10
```

Her flashcards even flash!

```
200 REM FLASHY FLASHCARDS
210 PRINT TAB(20);"RIGHT!";
220 FOR F=1 TO 20
230 C=INT(14*RND)+3
239 CALL SCREEN(2)
240 CALL SCREEN(C)
250 NEXT F
```

How about multiplication?

```
40 PRINT TAB(10);X;"X";Y;"=":::::::::
60 IF G=X*Y THEN 200
```

Or subtraction?

```
35 IF X<Y THEN 20
40 PRINT TAB(10);X;"-";Y;"=":::::::::
60 IF G=X-Y THEN 200
```

Or division?

```
20 X=INT(101*RND)+1
30 Y=INT(10*RND)+1
35 IF X<Y THEN 20
36 IF INT(X/Y)<>X/Y THEN 20
40 PRINT TAB(10);X;"/";Y;"=":::::::::
60 IF G=X/Y THEN 200
```

Main ideas: Creating a data list
Reading the data to print a selective list

Entering data lines takes a long time, and it's easy to make mistakes. Find a good friend, and work on this project together. You can check each other as you go along. Here are some tips:

1. Enter the data one line at a time, beginning at line 400. (Why 400? I'll tell you later!)
2. Count by 10s when you number the data lines (400, 410, 420, etc.).
3. Remember to start each data line with the word DATA.
4. Put a comma after each piece of data, but not at the end of the line.
5. Always enter the data in the same order: state, abbreviation, capital, flower, bird. For example:

```
4ØØ DATA ALABAMA,AL,MONTGOMERY,CAMELLIA,YELLOWHAMMER
```

6. Run the program after you enter each line to check for errors.

When you finish, you should have a listing of all 50 states. (If you get tired, you can stop after 10 or 20 states are entered. Add the rest next week when your typing fingers recover.) Later we will change the program so it does different things.

Here is an explanation of the basic program:

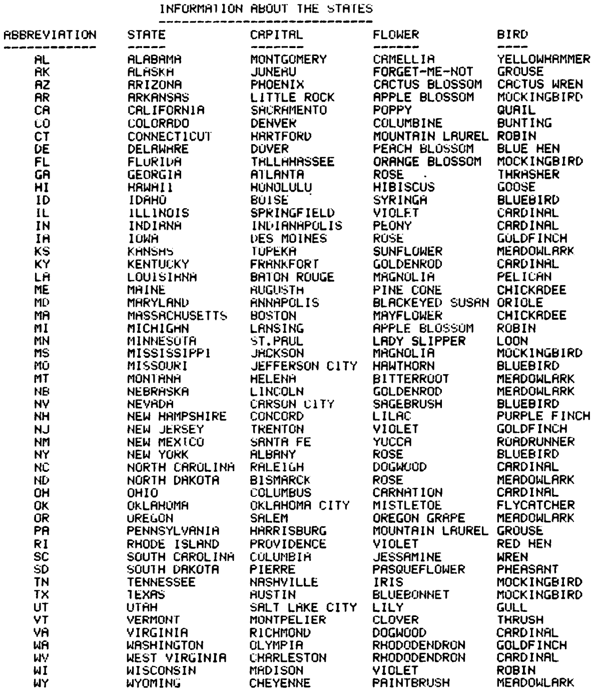
10 Clears the screen.
40 Reads the data in this order: ST$ (name of the state), AB$ (abbreviation), C$ (capital), F$ (flower), B$ (bird).
50 Checks for the flag data to end the program.
60 Prints the name of the state.
70 Goes back to line 40 to read more data.
400–999 The data lines that hold the information about each state. (Follow the pattern in line 400.)
1000 The flag data that signals the program to end. Line 40 is waiting to read 5 words, so write the word END 5 times.
1010 Ends the program.

To see a little of your list at a time, press FCTN–4 to stop the list as it scrolls up the screen. Then type "CONTINUE" to see a little more.

State the States

In social studies you are studying the United States. For extra credit, you and your best friend have volunteered to enter this data into your classroom computer. Good luck! (Go on to the next program to see how to use the data.)

```
10 CALL CLEAR
40 READ ST$,AB$,C$,F$,B$
50 IF ST$="END" THEN 1010
60 PRINT ST$
70 GOTO 40
400 DATA ALABAMA,AL,MONTGOMERY,CAMELLIA,YELLOWHAMMER

    (You add the rest of the data here!)

1000 DATA END,END,END,END,END
1010 END
```



```
                            INFORMATION ABOUT THE STATES
                            ----------------------------
ABBREVIATION    STATE           CAPITAL          FLOWER           BIRD
------------    -----           -------          ------           ----
     AL         ALABAMA         MONTGOMERY       CAMELLIA         YELLOWHAMMER
     AK         ALASKA          JUNEAU           FORGET-ME-NOT    GROUSE
     AZ         ARIZONA         PHOENIX          CACTUS BLOSSOM   CACTUS WREN
     AR         ARKANSAS        LITTLE ROCK      APPLE BLOSSOM    MOCKINGBIRD
     CA         CALIFORNIA      SACRAMENTO       POPPY            QUAIL
     CO         COLORADO        DENVER           COLUMBINE        BUNTING
     CT         CONNECTICUT     HARTFORD         MOUNTAIN LAUREL  ROBIN
     DE         DELAWARE        DOVER            PEACH BLOSSOM    BLUE HEN
     FL         FLORIDA         TALLAHASSEE      ORANGE BLOSSOM   MOCKINGBIRD
     GA         GEORGIA         ATLANTA          ROSE             THRASHER
     HI         HAWAII          HONOLULU         HIBISCUS         GOOSE
     ID         IDAHO           BOISE            SYRINGA          BLUEBIRD
     IL         ILLINOIS        SPRINGFIELD      VIOLET           CARDINAL
     IN         INDIANA         INDIANAPOLIS     PEONY            CARDINAL
     IA         IOWA            DES MOINES       ROSE             GOLDFINCH
     KS         KANSAS          TOPEKA           SUNFLOWER        MEADOWLARK
     KY         KENTUCKY        FRANKFORT        GOLDENROD        CARDINAL
     LA         LOUISIANA       BATON ROUGE      MAGNOLIA         PELICAN
     ME         MAINE           AUGUSTA          PINE CONE        CHICKADEE
     MD         MARYLAND        ANNAPOLIS        BLACKEYED SUSAN  ORIOLE
     MA         MASSACHUSETTS   BOSTON           MAYFLOWER        CHICKADEE
     MI         MICHIGAN        LANSING          APPLE BLOSSOM    ROBIN
     MN         MINNESOTA       ST.PAUL          LADY SLIPPER     LOON
     MS         MISSISSIPPI     JACKSON          MAGNOLIA         MOCKINGBIRD
     MO         MISSOURI        JEFFERSON CITY   HAWTHORN         BLUEBIRD
     MT         MONTANA         HELENA           BITTERROOT       MEADOWLARK
     NB         NEBRASKA        LINCOLN          GOLDENROD        MEADOWLARK
     NV         NEVADA          CARSON CITY      SAGEBRUSH        BLUEBIRD
     NH         NEW HAMPSHIRE   CONCORD          LILAC            PURPLE FINCH
     NJ         NEW JERSEY      TRENTON          VIOLET           GOLDFINCH
     NM         NEW MEXICO      SANTA FE         YUCCA            ROADRUNNER
     NY         NEW YORK        ALBANY           ROSE             BLUEBIRD
     NC         NORTH CAROLINA  RALEIGH          DOGWOOD          CARDINAL
     ND         NORTH DAKOTA    BISMARCK         ROSE             MEADOWLARK
     OH         OHIO            COLUMBUS         CARNATION        CARDINAL
     OK         OKLAHOMA        OKLAHOMA CITY    MISTLETOE        FLYCATCHER
     OR         OREGON          SALEM            OREGON GRAPE     MEADOWLARK
     PA         PENNSYLVANIA    HARRISBURG       MOUNTAIN LAUREL  GROUSE
     RI         RHODE ISLAND    PROVIDENCE       VIOLET           RED HEN
     SC         SOUTH CAROLINA  COLUMBIA         JESSAMINE        WREN
     SD         SOUTH DAKOTA    PIERRE           PASQUEFLOWER     PHEASANT
     TN         TENNESSEE       NASHVILLE        IRIS             MOCKINGBIRD
     TX         TEXAS           AUSTIN           BLUEBONNET       MOCKINGBIRD
     UT         UTAH            SALT LAKE CITY   LILY             GULL
     VT         VERMONT         MONTPELIER       CLOVER           THRUSH
     VA         VIRGINIA        RICHMOND         DOGWOOD          CARDINAL
     WA         WASHINGTON      OLYMPIA          RHODODENDRON     GOLDFINCH
     WV         WEST VIRGINIA   CHARLESTON       RHODODENDRON     CARDINAL
     WI         WISCONSIN       MADISON          VIOLET           ROBIN
     WY         WYOMING         CHEYENNE         PAINTBRUSH       MEADOWLARK
```

Main ideas: Using IF...THEN to find information on the data list
Using the data list to make a chart
Using the data list to play a game

Use the original program and add these extra lines. As you type the examples, try to figure out how the new lines fit into the old program. Can you make up some of your own examples?

To find information:

60 Tells what to look for after each group of data is read. Skips to line 80 if necessary.
80 Prints this message when line 60 is true.
90 Since you skipped line 70, this line goes back to line 40 to read more data.

To make a chart:

20 Prints the headings.
30 Underlines the headings. (There are several ways to underline.)
60 Tells the computer what to print.

To play a game:

60–61 Prints a question for you to answer.
62 Allows you to input your guess (A$).
63 Sends you to line 80 if your guess was correct.
64 Prints this message if your guess was incorrect.
80 Prints "RIGHT!" if your guess was correct.
90 If your guess was correct, you skipped line 70. You need line 90 to go back to line 40 to read more data.

To score the game:

85 Adds 1 to the number correct (C). By default, when you type "RUN" to begin the game, C is set to 0.
1010 Prints your score just before the program ends.
1020 Ends the program.

Trivia Expert

Let the computer turn you into a trivia expert. Who else could possibly know the state bird of Wyoming? Now that you put in all that data, let the computer help you:

To find information:

```
60 IF ST$="WYOMING" THEN 80
80 PRINT B$
90 GOTO 40
```

or

```
60 IF AB$="CT" THEN 80
80 PRINT C$
90 GOTO 40
```

or

```
60 IF F$="VIOLET" THEN 80
80 PRINT ST$
90 GOTO 40
```

or

```
60 IF B$="CARDINAL" THEN 80
80 PRINT ST$
90 GOTO 40
```

To make a chart:

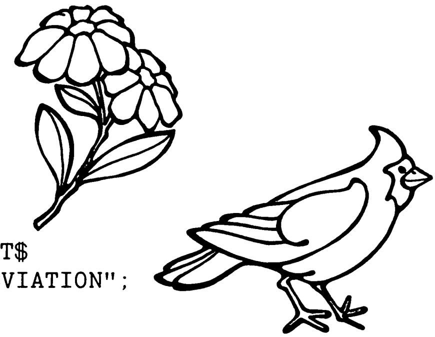
```
20 PRINT "STATE","CAPITAL"
30 PRINT "-----","-------"
60 PRINT ST$,C$
```

or

```
20 PRINT "STATE","BIRD"
30 PRINT "=====","===="
60 PRINT AB$,B$
```

or

```
20 PRINT "STATE","FLOWER"
30 PRINT "*****","******"
60 PRINT ST$,F$
```



To play a game:

```
60 PRINT "THE STATE IS ";ST$
61 PRINT "WHAT'S THE ABBREVIATION";
62 INPUT A$
63 IF A$=AB$ THEN 80
64 PRINT "WRONG, IT WAS ";AB$::
80 PRINT "RIGHT!"::
90 GOTO 40
```

To score the game:

```
85 C=C+1
1010 PRINT "YOUR SCORE IS ";C*2;"%"
1020 END
```

Main ideas: A race across the screen
Using ON...GOTO

10 Clears the screen.

20–50 Allows you to input your choices for the 3 worst lunches.

60 Clears the screen again.

70–100 Sets up the screen. Each of these lines prints something and then advances it up the screen by using colons.

110–130 Although the screen is 32 boxes across, the text begins at box 3. Therefore, all the lunches begin their race at box 3.

140 Ensures a random outcome.

150 Picks a number from 1 to 3 (a vote for the worst lunch).

160 The ON...GOTO statement saves you some IF...THEN statements. If the computer picked a 1, it goes to line 200. If it picked a 2, it goes to line 300. If it picked a 3, it goes to line 400.

200–230 This is what happens if the computer picked a 1:

200 Prints # (character 35) at box A right below the first lunch.

210 Adds 1 to A so the next # is printed next to the last #.

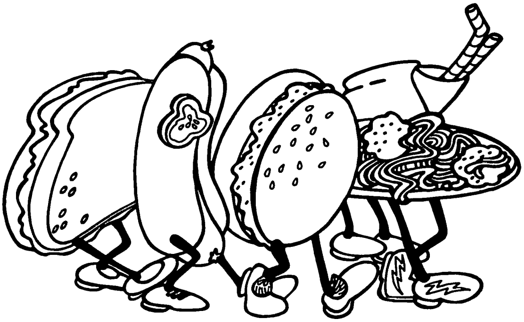
220 Checks to see if A is 33. Since the screen goes only to 32, if any lunch gets to 33, it must be a winner. If A=33, you go to line 500.

230 Otherwise, you go back to line 150. The computer picks the next lunch to get a vote.

300–330 This is what happens if the computer picked a 2.

400–430 This is what happens if the computer picked a 3.

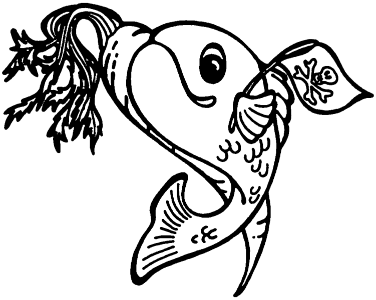
500 Prints this message when there is a winner—or should I say "loser"?

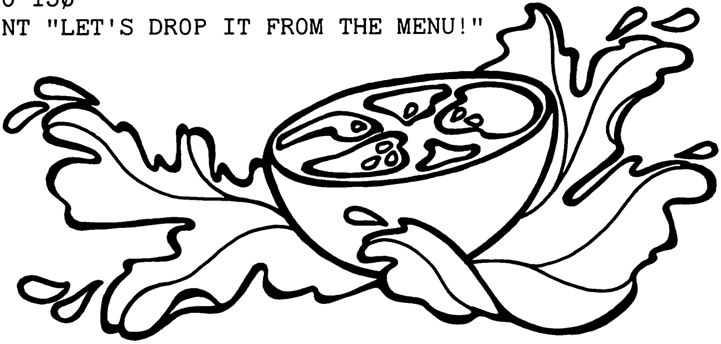


Gag Me with a Spoon

It's lunch time at last, but—oh no! It's the tuna salad boat *again!* What do *you* think is the worst school lunch?

```
1Ø CALL CLEAR
2Ø PRINT "THE 3 WORST LUNCHES ARE:"
3Ø INPUT A$
4Ø INPUT B$
5Ø INPUT C$
6Ø CALL CLEAR
7Ø PRINT "   THE WORST SCHOOL LUNCH"::::
8Ø PRINT A$:::::
9Ø PRINT B$:::::
1ØØ PRINT C$:::::
11Ø A=3
12Ø B=3
13Ø C=3
14Ø RANDOMIZE
15Ø X=INT(3*RND)+1
16Ø ON X GOTO 2ØØ,3ØØ,4ØØ
2ØØ CALL HCHAR(15,A,35)
21Ø A=A+1
22Ø IF A=33 THEN 5ØØ
23Ø GOTO 15Ø
3ØØ CALL HCHAR(2Ø,B,64)
31Ø B=B+1
32Ø IF B=33 THEN 5ØØ
33Ø GOTO 15Ø
4ØØ CALL HCHAR(25,C,42)
41Ø C=C+1
42Ø IF C=33 THEN 5ØØ
43Ø GOTO 15Ø
5ØØ PRINT "LET'S DROP IT FROM THE MENU!"
```





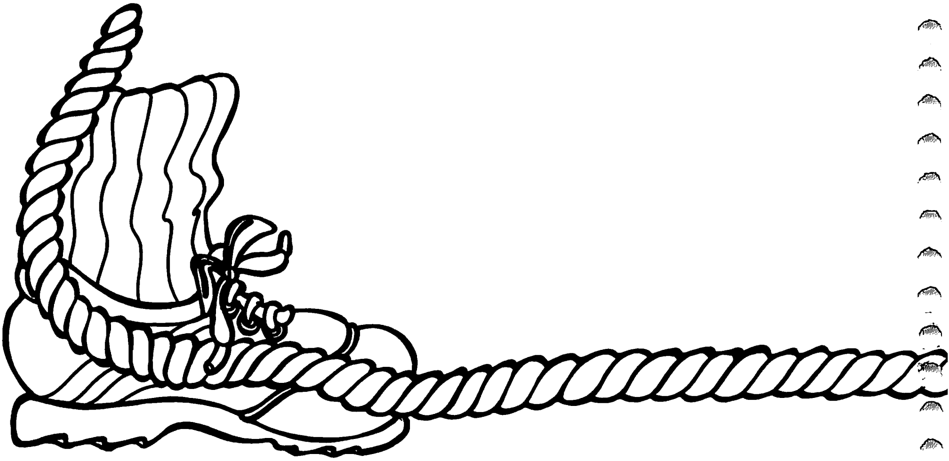
Main ideas: Inserting variables into a text
Using CALL KEY for "Press any key"

10 Clears the screen.
20–110 Allows you to input your ideas.
120 Clears the screen.
150–180 Prints the poem. Be careful with the spaces, semicolons, and colons. Remember that the semicolon (;) signals the computer to print a word right next to the last word printed. The colon (:) makes the computer skip a line.
190 By making the computer skip lines, we move the poem up to the middle of the screen.

200 Prints the direction.
210 Sets up the keyboard for instant input.
220 If you haven't pressed any key, you keep going back to line 210 until you do.
230 As soon as any key is pressed, you arrive at this line number, which sends you back to the beginning at line 10.

This section makes each verse appear in a different color:

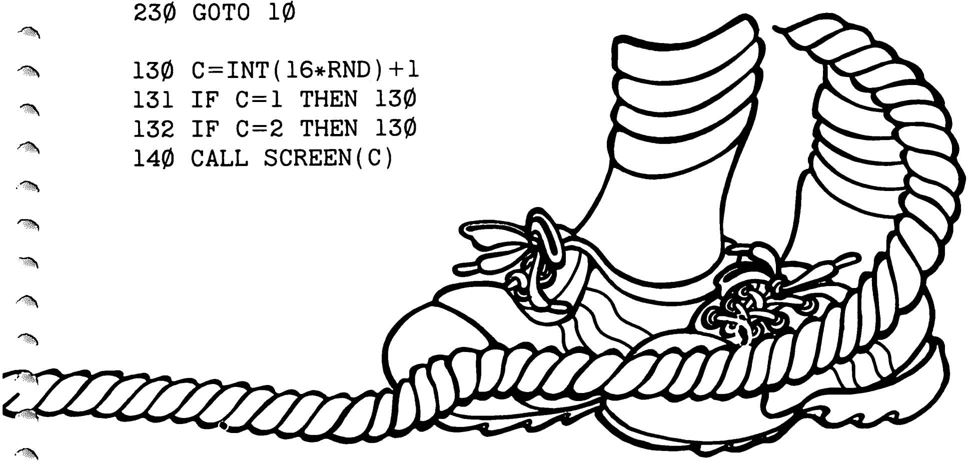
130 Picks a number from 1 to 16 for the color (C).
131–132 Screens that are transparent (color 1) or black (color 2) are disqualified because the black letters will not show up on these colors.
140 Otherwise, the screen becomes the color chosen.



Jump Rope

After lunch the girls like to jump rope. (I guess the boys don't know how!) This is one of their favorite poems. Why not make up some verses of your own?

```
1Ø CALL CLEAR
2Ø PRINT "LETTER OF ALPHABET";
3Ø INPUT L$
4Ø PRINT "GIRL'S NAME STARTING WITH ";L$
5Ø INPUT G$
6Ø PRINT "BOY'S NAME STARTING WITH ";L$
7Ø INPUT B$
8Ø PRINT "A PLACE THAT STARTS WITH ";L$
9Ø INPUT PL$
1ØØ PRINT "SOMETHING THAT BEGINS WITH ";L$;" (A
    PLURAL)";
11Ø INPUT TH$
12Ø CALL CLEAR
15Ø PRINT L$;",MY NAME IS ";G$;" AND"::
16Ø PRINT "MY HUSBAND'S NAME IS ";B$::
17Ø PRINT "WE COME FROM ";PL$::
18Ø PRINT "JUST TO SELL YOU ";TH$
19Ø PRINT ::::::

2ØØ PRINT "PRESS ANY KEY TO CONTINUE."
21Ø CALL KEY(Ø,N,STATUS)
22Ø IF STATUS=Ø THEN 21Ø
23Ø GOTO 1Ø

13Ø C=INT(16*RND)+1
131 IF C=1 THEN 13Ø
132 IF C=2 THEN 13Ø
14Ø CALL SCREEN(C)
```



Main ideas: Using CALL KEY for instant input from the keyboard
Drawing with the arrow keys

10 Clears the screen.

20 Sets the color of your path to blue (color 5). Use a different number from 3 to 16 if you like.

30 Randomizes the numbers. Otherwise, Joe's keys are located at the same place again and again.

40–50 Selects a hidden row (HR) and a hidden column (HC) for the location of the missing keys.

60–70 Sets your position (row and column) right in the middle of the screen.

80 Prints the message.

90 Prints your position (a blue box) on the screen.

100 If your row is the same as the hidden row, the program skips to line 250 to see if you're at the right column or not.

110 Sets up the keyboard for instant input.

120 If you haven't pushed any key, sends you back to line 110 until you do!

130–160 Checks to see which key you have pressed:
Character 69 is E, where the up arrow is located.
Character 88 is X, where the down arrow is located.
Character 68 is D, where the right arrow is located.
Character 83 is S, where the left arrow is located.

170–240 Adds 1 to, or subtracts 1 from, the row or the column of your position on the screen, depending on which arrow key you pressed.

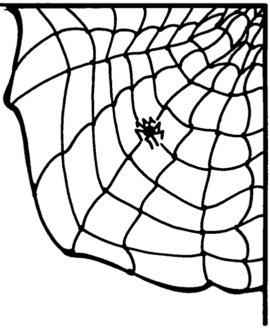
250 If your row is the same as the hidden row, you were sent to this line. If your column is different from the hidden column, you are sent back to line 110.

260 Otherwise, by deduction, you have found both the hidden row and the hidden column. Hooray for you!

270 Sets the program into an infinite loop for an ending. Press FCTN–4 to break out of the program.

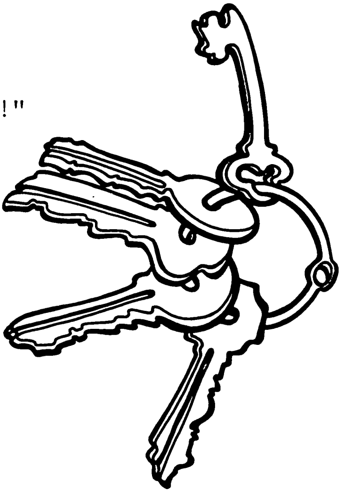
175–246 If you try to plot a point that is not on the screen, you get an error message. Reason your way through this section to see how it keeps you from going off the screen.

Joe's Keys



Joe, the custodian, is crawling around on his hands and knees. He has lost his keys somewhere. Poor Joe needs all the help he can get. Why don't you help Joe find his keys?

```
1Ø CALL CLEAR
2Ø CALL COLOR(2,5,5)
3Ø RANDOMIZE
4Ø HR=INT(24*RND)+1
5Ø HC=INT(32*RND)+1
6Ø R=12
7Ø C=16
8Ø PRINT "HELP JOE FIND HIS KEYS!"
9Ø CALL HCHAR(R,C,42)
1ØØ IF R=HR THEN 25Ø
11Ø CALL KEY(Ø,KEY,STATUS)
12Ø IF STATUS=Ø THEN 11Ø
13Ø IF KEY=69 THEN 17Ø
14Ø IF KEY=88 THEN 19Ø
15Ø IF KEY=68 THEN 21Ø
16Ø IF KEY=83 THEN 23Ø
17Ø R=R-1
18Ø GOTO 9Ø
19Ø R=R+1
2ØØ GOTO 9Ø
21Ø C=C+1
22Ø GOTO 9Ø
23Ø C=C-1
24Ø GOTO 9Ø
25Ø IF C<>HC THEN 11Ø
26Ø PRINT "YOU FOUND THEM!!"
27Ø GOTO 26Ø
```



```
175 IF R<1 THEN 185
185 R=1
186 GOTO 9Ø
195 IF R>24 THEN 2Ø5
2Ø5 R=24
2Ø6 GOTO 9Ø
215 IF C>32 THEN 225
225 C=32
226 GOTO 9Ø
235 IF C<1 THEN 245
245 C=1
246 GOTO 9Ø
```

Main ideas: A multiple-choice test

10 Clears the screen.

20–30 Prints the title and underlines it. TAB makes it count over to the 5th column before it starts to print.

40 Skips a line.

50–100 Each line prints 5 spaces and the name of an animal group.

110 Skips a line.

120 Prints the directions.

130 Skips a line.

140 Presents the questions, 3 at a time. This is how many questions fit on the screen if the choices are to remain visible.

150 Reads the name of an animal (AN$) and its classification number (CL). (Notice that the animal variable is a string variable and has a dollar sign ($), while the class variable is a numeric variable and has no dollar sign.)

160 When the word END is read, you are sent to line 280.

170 Skips a line.

180 Asks the question.

190 Allows you to input your guess (G).

200 Sends you to line 230 if your guess was correct (if G = CL).

210 Prints this message if your guess was incorrect and tells you the correct answer.

220 Skips to line 240 to avoid the "RIGHT!" message in line 230.

230 Prints "RIGHT!"

240 Keeps going back to line 140 until 3 questions are presented.

250–260 Pauses while the computer counts to 300. (Leave out these lines to see why you need them.)

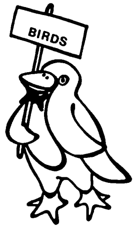
270 Goes back to line 10 each time 3 questions have been asked and answered.

280 Ends the program.

300–340 The data lines that hold the information for the test questions. Notice that they are arranged in the same pattern. (If you want to make the test longer, you may use lines 341 through 399 to add more data. If you choose to add more data, be sure to follow the pattern in lines 300 to 340.)

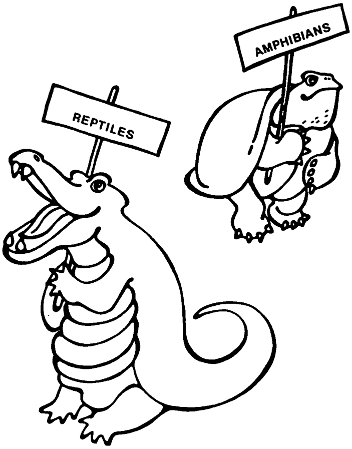
400 The flag data that signals the program to end (see line 160).

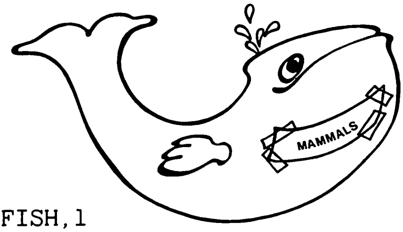
Science

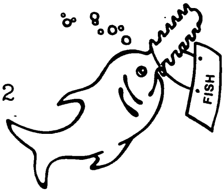


Lunch time is over.
The bell rings, and it's back to class you go. It's time for science. Did you know that all animals belong to one of these six groups?

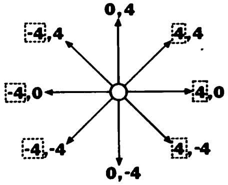
```
10 CALL CLEAR
20 PRINT TAB(5);"CLASSES OF ANIMALS"
30 PRINT TAB(5);"------------------"
40 PRINT
50 PRINT "     1.INVERTEBRATES"
60 PRINT "     2.FISH"
70 PRINT "     3.AMPHIBIANS"
80 PRINT "     4.REPTILES"
90 PRINT "     5.BIRDS"
100 PRINT "     6.MAMMALS"
110 PRINT
120 PRINT "PICK A NUMBER (1-6)";
130 PRINT
140 FOR Q=1 TO 3
150 READ AN$,CL
160 IF AN$="END" THEN 280
170 PRINT
180 PRINT "WHAT IS A ";AN$;
190 INPUT G
200 IF G=CL THEN 230
210 PRINT "WRONG,IT BELONGS TO GROUP ";CL
220 GOTO 240
230 PRINT "RIGHT!"
240 NEXT Q
250 FOR P=1 TO 300
260 NEXT P
270 GOTO 10
280 END
300 DATA DOG,6,TURTLE,4,STARFISH,1
310 DATA WHALE,6,TURKEY,5,FROG,3
320 DATA SALAMANDER,3,LIZARD,4,PEACOCK,5
330 DATA BAT,6,CLAM,1,SPIDER,1
340 DATA PERSON,6,HIPPOPOTAMUS,6,SALMON,2
400 DATA END,0
```







Main ideas: Using the joystick to draw on the screen
Using CALL KEY to change colors



To help you understand this program, first type in:

```
1 CALL KEY(Ø,KEY,STATUS)
2 PRINT KEY
3 GOTO 1
```

Find the Character Sets in the Appendix. Run the program with the ALPHA LOCK on, the way we usually do. You will get a string of "−1" until you press a key. When you press a key, its code appears on the screen. Can you figure out why you get a different number with the ALPHA LOCK off? (A small A has a different code from a big A!)

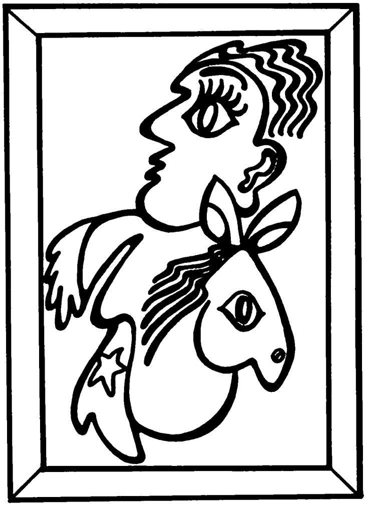
For the joystick to operate correctly, the ALPHA LOCK must come off before you run the program. During the program, then, we use the code numbers for small R (character 114 in set 11), small Y (character 121 in set 12), and small B (character 98 in set 9). Refer to lines 30, 40, and 50.

10–20 Clears the screen and makes it white (color 16).
30 Makes the characters in set 9 blue on blue (blue squares).
40 Makes the characters in set 11 red on red (red squares).
50 Makes the characters in set 12 yellow on yellow (yellow squares).
60–70 Sets the beginning row and column numbers at midscreen.
80 Sets the initial character at 114, the code for small R. This means that when you start the program, the computer draws in red.
90 Checks to see if you've pressed a key.
100 Goes to line 420.
110 Reads the joystick coordinates.
120–230 Adds 1 to or subtracts 1 from the row number (R) and/or the column number (C). This depends on which direction you pushed the joystick. Refer to the diagram and see if you can figure out how this section works.
300 A remark to save this line. Later we'll add lines 300 to 360. These lines will keep us on the screen by keeping the column number between 1 and 32 and the row number between 1 and 24.
400 Plots a red, blue, or yellow square at the current row and column. The color of the square depends on the character number.
410 Goes back to line 90 to draw another square.
420–450 The GOSUB routine for line 100:
430 If you didn't press a key, it skips to line 450.
440 If you pressed a key, the code for the key becomes the number for the character to be drawn.
450 Returns to the main program.

Note: When you press FCTN–4 to break out of the program, the colors disappear and you will see all the little Rs, Ys, and Bs.

Modern Art

The art teacher is absent, and
the sub lets you draw whatever you like.
Press R for red, Y for yellow, and
B for blue. Maybe you can exhibit
your work at the Modern
Art Museum!



```
1Ø CALL CLEAR
2Ø CALL SCREEN(16)
3Ø CALL COLOR(9,5,5)
4Ø CALL COLOR(11,9,9)
5Ø CALL COLOR(12,12,12)
6Ø C=16
7Ø R=12
8Ø CHAR=114
9Ø CALL KEY(Ø,KEY,STATUS)
1ØØ GOSUB 42Ø
11Ø CALL JOYST(1,X,Y)
12Ø IF X=-4 THEN 15Ø
13Ø IF X=4 THEN 17Ø
14Ø GOTO 18Ø
15Ø C=C-1
16Ø GOTO 18Ø
17Ø C=C+1
18Ø IF Y=-4 THEN 21Ø
19Ø IF Y=4 THEN 23Ø
2ØØ GOTO 3ØØ
21Ø R=R+1
22Ø GOTO 3ØØ
23Ø R=R-1
3ØØ REM
4ØØ CALL HCHAR(R,C,CHAR)
41Ø GOTO 9Ø
42Ø REM CHANGES COLOR
    IF NECESSARY
43Ø IF KEY=-1 THEN 45Ø
44Ø LET CHAR=KEY
45Ø RETURN
```

```
3ØØ REM KEEP ON CANVAS
31Ø IF R>24 THEN 32Ø
311 IF R<1 THEN 33Ø
312 GOTO 34Ø
32Ø R=24
321 GOTO 34Ø
33Ø R=1
34Ø IF C>32 THEN 35Ø
341 IF C<1 THEN 36Ø
342 GOTO 4ØØ
35Ø C=32
351 GOTO 4ØØ
36Ø C=1
```

Main ideas: Using CALL KEY for instant input from the keyboard ASCII codes
Converting keystrokes to musical notes

10 Clears the screen.

20 Prints the direction. The colons advance it up the screen.

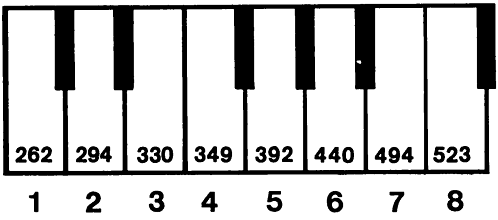
30 Sets up the keyboard for instant input. As soon as you press one of the desired keys, a note will play.

40 If you haven't pressed any key, you keep going back to line 30 until you do!

41 If you leave out this line, the note may play repeatedly when you wanted to hear it only once. These repetitions occur because you don't take your hand off the key fast enough. (Leave out this line and see what I mean!) The "−1" status means that the same key has been pressed.

50 Refer to the Character Sets in the Appendix. You will see that character 49 is 1, character 50 is 2, and so forth. If you press a 1, you are sent to line 150, and the note becomes 262 (middle C). Then, in line 151, you are sent to line 300, which plays the note.

60–211 See if you can trace this section of the program. It is responsible for assigning notes to each key pressed. The octave used in the program is illustrated below. For a more complete keyboard, refer to the Musical Notes in the Appendix.



300 Plays each note for 300 milliseconds.

310 Goes to line 30 to see what you press next.

These lines change the background color for each note played:

305 Picks a number from 1 to 16 for the color (C).

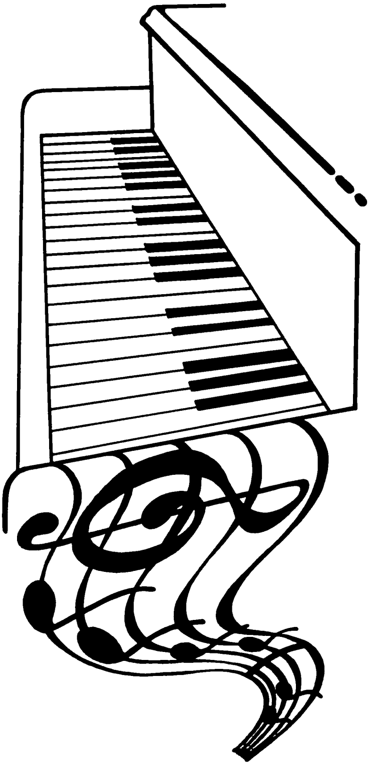
306–307 Transparent (color 1) and black (color 2) are disqualified because the print will not show up.

308 Makes the screen the color chosen.

Music

The music teacher has asked everyone to compose a tune. It's time to play your song. Do a good job!

```
1Ø CALL CLEAR
2Ø PRINT "TO MAKE MUSIC PRESS KEYS 1-8"::::::::::
3Ø CALL KEY(Ø,K,STATUS)
4Ø IF STATUS=Ø THEN 3Ø
41 IF STATUS=-1 THEN 3Ø
5Ø IF K=49 THEN 15Ø
6Ø IF K=5Ø THEN 16Ø
7Ø IF K=51 THEN 17Ø
8Ø IF K=52 THEN 18Ø
9Ø IF K=53 THEN 19Ø
1ØØ IF K=54 THEN 2ØØ
11Ø IF K=55 THEN 21Ø
12Ø N=523
121 GOTO 3ØØ
15Ø N=262
151 GOTO 3ØØ
16Ø N=294
161 GOTO 3ØØ
17Ø N=33Ø
171 GOTO 3ØØ
18Ø N=349
181 GOTO 3ØØ
19Ø N=392
191 GOTO 3ØØ
2ØØ N=44Ø
2Ø1 GOTO 3ØØ
21Ø N=494
211 GOTO 3ØØ
3ØØ CALL SOUND(3ØØ,N,2)
31Ø GOTO 3Ø
```



You'll get extra credit
for an interesting performance.

```
3Ø5 C=INT(16*RND)+1
3Ø6 IF C=1 THEN 3Ø5
3Ø7 IF C=2 THEN 3Ø5
3Ø8 CALL SCREEN(C)
```

Main ideas: Averaging a list of numbers
Using CALL KEY for "Press any key"

10 Clears the screen.
20–30 Allows you to input the student's name.
40 Skips a line.
50–60 Prints the directions.
70 The total number of tests (T) is set to 0.
80 The sum of the test scores (S) is set at 0.
90 Allows you to input a test score (SC).
100 If you type a "−1," you are sent to line 140 immediately.
110 The number of tests goes up 1.
120 The sum of the scores (S) is increased by whatever number the score (SC) was. For example, if the sum is 0 and you score 90 on the first test, the new sum is now 90. If you score 90 on the second test, the sum becomes 180, and so forth.
130 Goes back to line 90 so you can enter the next test score.
140 A remark.
150 Computes the average. The average (AV) is the sum of the scores (S), divided by the number of tests (T). For example, if the sum of all the test scores is 450 and you took 5 tests, your average is 90.
160 Prints the average.
170–190 Evaluates your average, and gives you a grade by sending you to an appropriate line number.
200 If your score was below 70, you are automatically given a U (for unsatisfactory).
210–270 Prints your grade and sends you to line 280.
280 Ends the program

This program works just fine for a single student, but if you have several students, you might like to add these lines so you don't have to keep typing the word RUN.

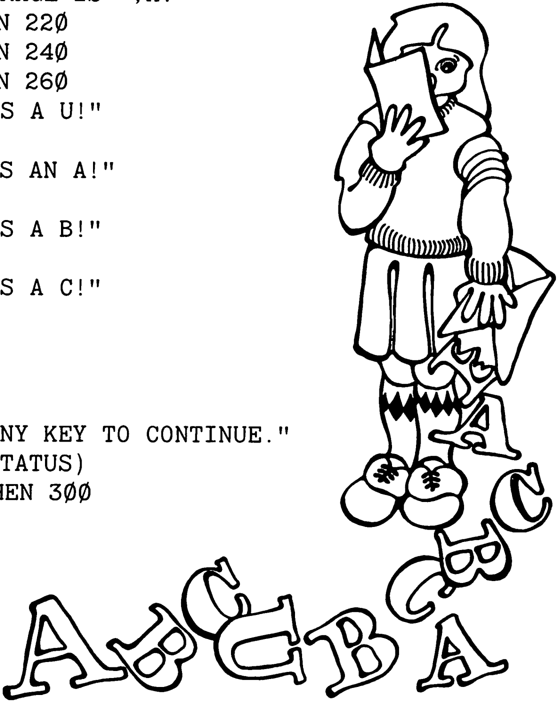
280 Skips a line.
290 Prints the direction.
300 Sets up the keyboard for instant input.
310 If you haven't pressed any key, you keep going back to line 300 until you do!
320 As soon as you press a key, you go back to line 10 to start over with a new student.

If you add lines 280 to 320, you will create an "infinite loop" situation. You must type FCTN–4 to break out of the program.

Report Card

Today is report card day. Average your test grades, and find out what you got on your report card.

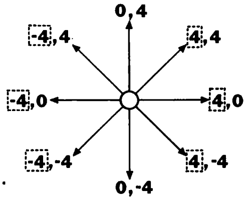
```
1Ø CALL CLEAR
2Ø PRINT "NAME OF STUDENT:"
3Ø INPUT N$
4Ø PRINT
5Ø PRINT "SCORES ON TESTS:"
6Ø PRINT "(TYPE -1 WHEN FINISHED)"
7Ø T=Ø
8Ø S=Ø
9Ø INPUT SC
1ØØ IF SC=-1 THEN 14Ø
11Ø T=T+1
12Ø S=S+SC
13Ø GOTO 9Ø
14Ø REM FIND AVERAGE
15Ø AV=S/T
16Ø PRINT "THE AVERAGE IS ";AV
17Ø IF AV>=9Ø THEN 22Ø
18Ø IF AV>=8Ø THEN 24Ø
19Ø IF AV>=7Ø THEN 26Ø
2ØØ PRINT N$;" GETS A U!"
21Ø GOTO 28Ø
22Ø PRINT N$;" GETS AN A!"
23Ø GOTO 28Ø
24Ø PRINT N$;" GETS A B!"
25Ø GOTO 28Ø
26Ø PRINT N$;" GETS A C!"
27Ø GOTO 28Ø
28Ø END

28Ø PRINT
29Ø PRINT "PRESS ANY KEY TO CONTINUE."
3ØØ CALL KEY(Ø,N,STATUS)
31Ø IF STATUS=Ø THEN 3ØØ
32Ø GOTO 1Ø
```



Main Ideas: Scrolling
Using the joystick

When you type something on the screen, everything previously typed scrolls up the screen. If there is something printed at the top of the screen, it scrolls right *off* the screen. This program takes advantage of the scrolling action.

10 Clears the screen.
20 The position of the bicycle (B) is set at 16, which is halfway across the screen.
30 Character 50 in set 3 is being used for the bicycle (really a square).
40 The number of loops this program makes is set to 100.
50 Picks a number from 1 to 32 for the position of the car (C).
60 Picks a number from 1 to 32 for the position of the truck (T).
70 Reads the joystick numbers for X and Y.
80 Goes to line 175 to interpret the joystick reading.
90 Prints a colon (:) for the truck at row 23 near the bottom of the screen.
100 Prints an @ symbol for the car at row 24, the last row on the screen. (You can change the symbols for the car and the truck by referring to the Character Sets in the Appendix and changing the character numbers.)
110 Draws a bicycle about halfway up the screen at row 12.
120 This line is very important. It prints a blank line so the whole screen scrolls up.
130 Goes back to line 40 for the next loop.
140–170 Prints an ending; ends the program.
175 A reminder that this section reads the joystick.
180 If you push the joystick to the left, the X coordinate is −4 (see the diagram). You go to line 210.



190 If you push the joystick to the right, the X coordinate is 4 (see the diagram). You go to line 260.
210 Subtracts 1 from the position of the bicycle.
220 Checks to see if you're about to go off the screen.
230 If you're not, you go to line 300.
240 If you are, the bicycle is kept at position 1 and line 250 sends you to line 300.
260 Adds 1 to the position of the bicycle.
270 If you're about to go off the screen, the bicycle is kept at position 32 (line 290), and finally you are sent to line 300.
300 Returns to the main part of the program to put the bicycle at the proper position on the screen.

The Ride Home

Your Mom is really going to like your report card. She'll probably make your favorite after-school treat. You decide to race home on your bicycle. Sometimes you even race against traffic. Watch out!

```
1Ø CALL CLEAR
2Ø B=16
3Ø CALL COLOR(3,5,5)
4Ø FOR N=1 TO 1ØØ
5Ø C=INT(32*RND)+1
6Ø T=INT(32*RND)+1
7Ø CALL JOYST(1,X,Y)
8Ø GOSUB 175
9Ø CALL HCHAR(23,T,58)
1ØØ CALL HCHAR(24,C,64)
11Ø CALL HCHAR(12,B,5Ø)
12Ø PRINT
13Ø NEXT N
14Ø PRINT "YOU'RE HOME!"
15Ø PRINT "HOW MANY FENDER BENDERS"
16Ø PRINT "   DID YOU HAVE?"
17Ø END
175 REM READING THE JOYSTICK
18Ø IF X=-4 THEN 21Ø
19Ø IF X=4 THEN 26Ø
2ØØ GOTO 3ØØ
21Ø B=B-1
22Ø IF B<1 THEN 24Ø
23Ø GOTO 3ØØ
24Ø B=1
25Ø GOTO 3ØØ
26Ø B=B+1
27Ø IF B>32 THEN 29Ø
28Ø GOTO 3ØØ
29Ø B=32
3ØØ RETURN
```

Main ideas: Counting backwards
Using flag data
RESTORE-ing data to be used again

10 Begins a FOR...NEXT loop that will count backwards from 99 to 1. (The end of the loop is at line 120.)

20 Clears the screen.

30–40 Prints the first line of the song.

50 Reads the note (N) and the time (T) that the note gets. The time is given as number of beats.

60–90 Each line here checks to see if flag data has been read. Look at lines 280 to 310, noticing the data at the end of each line. Each number signals the computer to go to a certain line of the program, which directs it to print the next line of the song on the screen.

100 Plays the note for 100 milliseconds per beat. (Less than 100 milliseconds, the song goes faster; more than 100, it goes slower.)

110 Goes back to line 50 to read the next note.

120 Finishes the loop, and sends you back for more.

130–200 On cue the computer prints the lines of the song, one at a time. Then it goes back to line 50 to read more notes.

210 Restores the data so it can be read again for the next verse of the song.

220 Advances the verse up the screen.

230 If there is more than one bottle left on that wall, you go back to line 120 for another verse.

240–260 Prints these lines when the song is over.

270 A "dummy" line is used to freeze the screen. (Type FCTN–4 to get out of this infinite loop.)

280–310 The data lines that hold the information for the program. (Refer to the Musical Notes in the Appendix if you want to see where the numbers come from.)

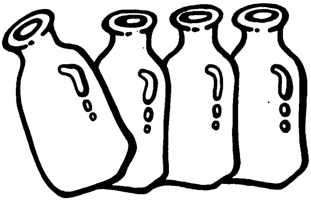
This little addition makes the screen change color for each verse.

21 Randomizes the numbers.

22 Picks a number from 1 to 16 for the color.

23–24 If transparent (color 1) or black (color 2) is chosen, you are sent back to line 22 for another color selection. (The letters will not show up on a black or transparent screen.)

25 Makes the screen the color chosen.



Field Trip

Today's the day of the big game. Your school is playing an arch rival. The school has provided busses to get to the game. On the bus, the kids sing this old classic.

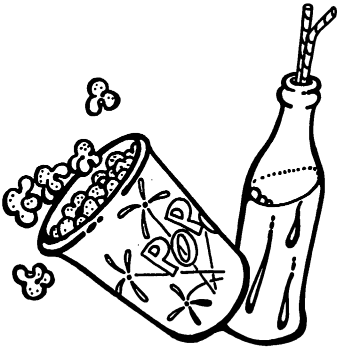
```
1Ø FOR B=99 TO 1 STEP-1
2Ø CALL CLEAR
3Ø PRINT B;"BOTTLES OF MILK"
4Ø PRINT "   ON THE WALL"::
5Ø READ N,T
6Ø IF N=-1 THEN 13Ø
7Ø IF N=-2 THEN 15Ø
8Ø IF N=-3 THEN 18Ø
9Ø IF N=-4 THEN 21Ø
1ØØ CALL SOUND(1ØØ*T,N,2)
11Ø GOTO 5Ø
12Ø NEXT B
13Ø PRINT B;"BOTTLES OF MILK-"::
14Ø GOTO 5Ø
15Ø PRINT "   TAKE ONE DOWN AND"
16Ø PRINT "    PASS IT AROUND"::
17Ø GOTO 5Ø
18Ø PRINT B-1;"BOTTLES OF MILK"
19Ø PRINT "   ON THE WALL!"
2ØØ GOTO 5Ø
21Ø RESTORE
22Ø PRINT ::::::
23Ø IF B>1 THEN 12Ø
24Ø PRINT "HOW CAN THE BUSDRIVER DRIVE"
25Ø PRINT "WITH HIS FINGERS IN HIS EARS"
26Ø PRINT "????????????????????????????"
27Ø GOTO 27Ø
28Ø DATA 349,1,349,1,349,1,349,1,
    349,1,349,1,349,1,349,1,349,1,
    349,3,-1,-1
29Ø DATA 392,1,392,1,392,1,294,1,
    294,1,294,1,392,6,-2,-2
3ØØ DATA 33Ø,3,33Ø,1,33Ø,3,33Ø,1,
    33Ø,1,33Ø,1,33Ø,1,33Ø,3,-3,-3
31Ø DATA 262,1,262,1,262,1,294,1,
    294,1,33Ø,1,349,1,349,1,349,1,
    349,3,-4,-4
```

Why not change the color for each verse?

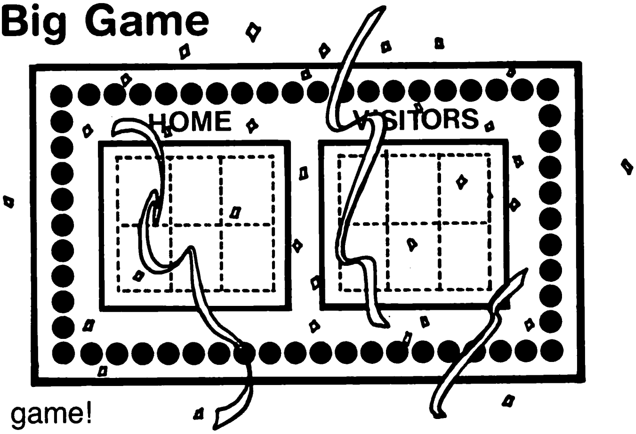
```
21 RANDOMIZE
22 C=INT(16*RND)+1
23 IF C=1 THEN 22
24 IF C=2 THEN 22
25 CALL SCREEN(C)
```

Main ideas: Random outcome of a game
Incrementing a total

10 Clears the screen.
20–50 Allows you to input the team names.
60 Clears the screen.
70 Makes the screen cyan (color 8). (Refer to the Colors in the Appendix to select a different color.)
80–90 Counts over to the 5th column and prints the title.
100 Skips a line.
110–120 At the beginning of the game, your team (Y) has no points and their team (TH) has no points.
130 The game keeps coming out the same way unless you use this line.
140 Since there's a basket a minute during the game, this loop will repeat 60 times.
150 Picks a number for the basket (B), a 0 or a 1.
160 If it picked a 0, you skip to line 190 where your team gets 2 points added to its total.
170 Otherwise, their team gets the 2 points and line 180 sends you to line 200.
200 Prints the score. Notice the difference between the variable names. Y$ stands for the name of the team; it is a string variable. Y stands for the number of points your team scored; it is a numeric variable and has no dollar sign ($). (The same applies to TH$, which is a string variable, and TH, which is a numeric variable.)
210 Goes back to line 140 for the next "minute" of the game.
220 Skips a line.
230 If the score is tied, you are sent to line 260.
240 If the score is uneven, somebody won, and this line is printed.
250 Sends you to the end of the program at line 270.
260 Prints this message when the score is tied.
270 Ends the program.



The Big Game



What a game!
There was a basket scored every minute.
Here's how the game turned out:

```
10 CALL CLEAR
20 PRINT "NAME OF YOUR TEAM";
30 INPUT Y$
40 PRINT "NAME OF THEIR TEAM";
50 INPUT TH$
60 CALL CLEAR
70 CALL SCREEN(8)
80 PRINT TAB(5);"THE BIG GAME"
90 PRINT TAB(5);"------------"
100 PRINT
110 Y=0
120 TH=0
130 RANDOMIZE
140 FOR M=1 TO 60
150 B=INT(2*RND)
160 IF B=0 THEN 190
170 TH=TH+2
180 GOTO 200
190 Y=Y+2
200 PRINT Y$;Y,TH$;TH
210 NEXT M
220 PRINT
230 IF Y=TH THEN 260
240 PRINT TAB(10);"FINAL!!"
250 GOTO 270
260 PRINT TAB(10);"REMATCH!"
270 END
```

Main ideas: Making a chart
Using TAB, commas, and semicolons to format printouts
Using formulas to compute results

The TAB command counts over to a certain column on the screen before printing. The semicolon prints one piece of information right next to another. The comma sends the cursor halfway across the screen to the next print zone. These three devices were used throughout this program.

10 Clears the screen.

20–50 Allows you to choose your pizza and tell how much it costs.

60 Clears the screen again.

70 Counts over to column 5 and prints the kind of pizza (K$).

80 Counts over to column 3 and underlines the kind of pizza.

90 Prints the price of the pizza (P).

100 Figures the tip (TP) as 15% of the price.

110 Prints the tip at column 14, next to the price.

120 Figures the new price as the old price plus the tip.

130 Prints the total price (including the tip).

140 Skips a line.

150–180 Prints and underlines the headings for the chart.

190–250 Does the figuring for 1 to 8 persons:

200 The cost per person (C) is the total price (P), divided by the number of persons (N).

210 The number of slices per person (SL) is the number of slices in the pie (8), divided by the number of persons (N).

220 Prints the number of persons. The semicolon holds the cursor at the present row and waits for the next piece of information to be printed.

230 Counts to the 8th column in the row and prints the cost per person (C).

240 Prints the number of slices per person (SL) at the 16th column.

250 Goes back to line 190 to repeat the loop for the next number of persons.

260 Moves the chart up the screen.

270 A dummy line is used to freeze the chart in position on the screen.

The extra lines (lines 101, 201, and 211) are used to keep you from getting long decimal numbers in the chart. Lines 101 and 201 round off the amount of the tip and the cost per person to the nearest hundredth. This formula is often used to round off money to the nearest penny. Line 211 rounds off the number of slices to tenths—about as close as you would care to estimate when dividing a piece of pizza!

Pizza

After the game, you and some friends go out for pizza. It's hard to decide what to get. How much will it cost? How much is the tip? How much should each person chip in? How much may each person eat? Let the computer do your arithmetic.

MENU

Plain	*6.95*
Extra Cheese	*7.95*
Sausage	*8.95*
Pepperoni	*8.95*
Meatball	*8.95*
Peppers	*7.95*
Onions	*7.95*
The Works	*9.95*

All pies: 8 slices

FREE Pitcher of Soda with Every Pizza Order

```
1Ø CALL CLEAR
2Ø PRINT "WHAT KIND DO YOU WANT?"
3Ø INPUT K$
4Ø PRINT "HOW MUCH DOES IT COST";
5Ø INPUT P
6Ø CALL CLEAR
7Ø PRINT TAB(5);K$;" PIZZA"
8Ø PRINT TAB(3);"-------------------"
9Ø PRINT "COST: ";P;
1ØØ TP=.15*P
11Ø PRINT TAB(14);"TIP: ";TP
12Ø P=P+TP
13Ø PRINT " TOTAL COST: ";P
14Ø PRINT
15Ø PRINT "NO.OF";
16Ø PRINT TAB(9);"COST/"," SLICES/"
17Ø PRINT "PERSONS PERSON PERSON"
18Ø PRINT "--------------------------"
19Ø FOR N=1 TO 8
2ØØ C=P/N
21Ø SL=8/N
22Ø PRINT N;
23Ø PRINT TAB(8);C;
24Ø PRINT TAB(16);SL
25Ø NEXT N
26Ø PRINT :::
27Ø GOTO 27Ø

1Ø1 TP=INT(1ØØ*TP+.5)/1ØØ
2Ø1 C=INT(1ØØ*C+.5)/1ØØ
211 SL=INT(10*SL+.5)/1Ø
```

Main ideas: Keeping a running total during a game of chance

5–20 Gets the game started.

25 A remark to remind you what this section is about.

30–35 Clears the screen and makes it blue (color 5).

40 Because there will be writing on the screen, the boxes will be drawn using a character not used for printing messages. I chose small A (character 97 in set 9). This line sets all characters in set 9 to red on red, or red squares.

45–65 Draws the 3 boxes at rows 7 to 12:

50 The first box starts at column 5 and extends for 5 boxes.

55 The second box starts at column 13 and extends for 5 boxes.

60 The third box starts at column 21 and extends for 5 boxes.

During the game, the total (T) is decreased by 25¢ each time you play. It is increased by 50¢ each time you win a game.

100 Another remark.

105 Randomizes the numbers.

110 Picks a number (M) from 1 to 3, where the money is hidden.

115 Subtracts 25¢ from your total so you can play.

120 Prints how much money you have left (your new total).

125 If you have 0 or less, you don't have enough money to play—you skip to line 300. (Bye!)

130 Asks you to choose a box.

135 Allows you to input your guess (G).

140 Sends you to line 200 if your guess was correct.

145 Goes back to line 25 to give you a chance to lose more money.

200 A remark.

201 Adds 50¢ to your total.

205–225 A repeating loop to make flashes:

210 Prints a blank line, which moves the boxes up the screen.

215 Picks a number from 0 to 13, and adds 3 to that number. (This effectively eliminates colors 1 and 2, which would make it impossible to read messages on the screen.)

220 Makes the screen the color chosen in line 215.

221 Prints your new total.

225 Goes back to line 205 for the next flash.

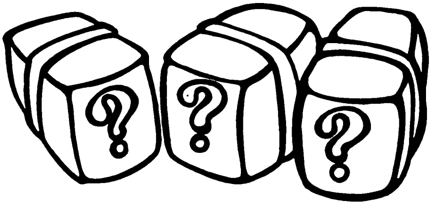
230 When the flashing is finished, goes back to line 25 to start again.

300–310 Ends the game.

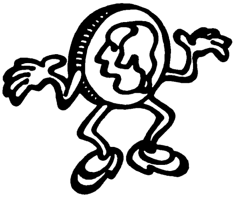
Double or Nothing

"Double or Nothing" is your favorite game at the school fair. The person at the booth has three little boxes. A half dollar is hiding in one of them. Guess which one.

```
5 CALL CLEAR
6 PRINT "HOW MUCH MONEY DO YOU HAVE?"
7 INPUT T
1Ø PRINT "YOU MAY CHOOSE BOX 1,2 OR 3."
15 PRINT "ARE YOU READY?"
2Ø INPUT R$
25 REM DRAW BOXES
3Ø CALL CLEAR
35 CALL SCREEN(5)
4Ø CALL COLOR(9,9,9)
45 FOR R=7 TO 12
5Ø CALL HCHAR(R,5,97,5)
55 CALL HCHAR(R,13,97,5)
6Ø CALL HCHAR(R,21,97,5)
65 NEXT R

1ØØ REM GAME
1Ø5 RANDOMIZE
11Ø M=INT(3*RND)+1
115 T=T-.25
12Ø PRINT "YOU NOW HAVE $";T
125 IF T<=Ø THEN 3ØØ
13Ø PRINT "WHICH DO YOU CHOOSE";
135 INPUT G
14Ø IF G=M THEN 2ØØ
145 GOTO 25
2ØØ REM YOU WON
2Ø1 T=T+.5Ø
2Ø5 FOR F=1 TO 1Ø
21Ø PRINT
215 C=INT(14*RND)+3
22Ø CALL SCREEN(C)
221 PRINT TAB(5);"YOU'RE UP TO $";T
225 NEXT F
23Ø GOTO 25
3ØØ REM END OF GAME
3Ø5 CALL CLEAR
31Ø PRINT TAB(4);"SORRY YOU ARE BROKE!!"::::::::::
```



You pay 25¢ to play. If you pick the right box, you keep the half dollar. You may play until you go broke!



Main ideas: Using READ, DATA, and RESTORE
Incrementing a total

This program finds the number of days between any two dates that you select. It does so by converting each date to its numerical position in the year. For example, January 1 is day 1 in the year, January 31 is day 31, February 1 is day 32, and December 31 is day 365. The program converts each date by adding up the number of days in each month preceding a given date and then adding on the number of days in the given month. To find out what day April 10th is, we reason like this:

$$\text{April 10 or 4/10} = \underbrace{31 + 28 + 31}_{\text{days in the 3 months preceding April}} + \underset{\text{days in April}}{10} = 100$$

If the month is January (which is the first month), you don't have to add up any months. Just use the number of the day. For example, January 13 is day 13. Now let's look at the program lines individually:

10 Clears the screen.

20–30 Initially, sets day 1 and day 2 at 0.

40–80 Allows you to input the beginning date—month and day. (Enter numbers for the months—1 for January, 2 for February, etc.)

90 If the month is January, you skip to line 150.

100 Counts to the number just before the given month. (If the month is 5, for example, it counts to 4; that is, if the month is May, it will add up the number of days in the first 4 months.)

110 Reads the number of days in each month (data line 500).

120 Adds the number of days to day 1.

130 Finishes the loop begun in line 100.

140 Restores the data, which will be read again to figure day 2.

150 Adds the number of the day in the current month to the total days in the previous months.

160 Prints the number of day 1.

170–290 Figures the number of day 2.

300 Prints 2 blank lines.

310 Finds the number of days between day 1 and day 2.

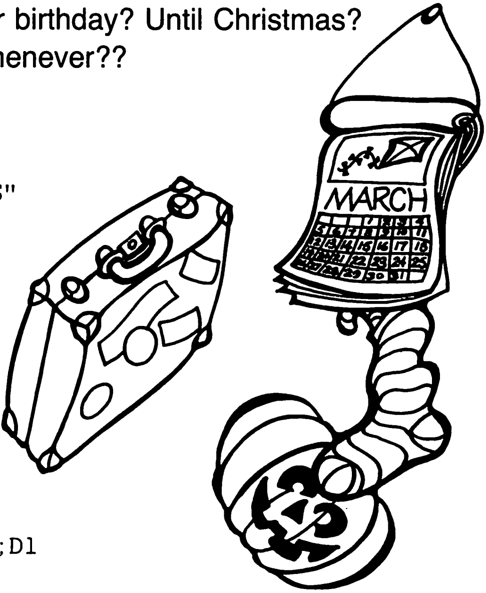
320–340 Fixes things up when you go over the first day of the year. For instance, without this line, the computer would say that there are −364 days from December 31 to January 1!

350 Prints the results.

500 The data line that holds the number of days in each month.

Days to Go

No matter how much fun school is, everyone likes to figure out how many days until summer vacation. While you're at it, how many days until your birthday? Until Christmas? Until Halloween? Until whenever??



```
1Ø CALL CLEAR
2Ø D1=Ø
3Ø D2=Ø
4Ø PRINT "HOW MANY DAYS"
5Ø PRINT "FROM MONTH";
6Ø INPUT M
7Ø PRINT "      DAY";
8Ø INPUT D
9Ø IF M=1 THEN 15Ø
1ØØ FOR X=1 TO M-1
11Ø READ N
12Ø LET D1=D1+N
13Ø NEXT X
14Ø RESTORE
15Ø LET D1=D1+D
16Ø PRINT "DAY 1 IS #";D1

17Ø PRINT : :
18Ø PRINT "TO MONTH";
19Ø INPUT M
2ØØ PRINT "    DAY";
21Ø INPUT D
22Ø IF M=1 THEN 28Ø
23Ø FOR X=1 TO M-1
24Ø READ N
25Ø LET D2=D2+N
26Ø NEXT X
27Ø RESTORE
28Ø LET D2=D2+D
29Ø PRINT "DAY 2 IS #";D2

3ØØ PRINT : :
31Ø T=D2-D1
32Ø IF T<Ø THEN 34Ø
33Ø GOTO 35Ø
34Ø T=T+365
35Ø PRINT "THERE ARE";T;"DAYS TO GO"
5ØØ DATA 31,28,31,3Ø,31,3Ø,31,31,3Ø,31,3Ø,31
```

APPENDIX

Character Sets

Set #1		Set #2		Set #3		Set #4		Set #5		Set #6	
Code	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*
32	(space)	40	(	48	0	56	8	64	@	72	H
33	!	41	)	49	1	57	9	65	A	73	I
34	”	42	*	50	2	58	:	66	B	74	J
35	#	43	+	51	3	59	;	67	C	75	K
36	$	44	,	52	4	60	<	68	D	76	L
37	%	45	−	53	5	61	=	69	E	77	M
38	&	46	.	54	6	62	>	70	F	78	N
39	’	47	/	55	7	63	?	71	G	79	O

Set #7		Set #8		Set #9		Set #10		Set #11		Set #12	
Code	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*	*Code*	*Character*
80	P	88	X	96	`	104	H	112	P	120	X
81	Q	89	Y	97	A	105	I	113	Q	121	Y
82	R	90	Z	98	B	106	J	114	R	122	Z
83	S	91	[	99	C	107	K	115	S	123	{
84	T	92	\	100	D	108	L	116	T	124	¦
85	U	93	]	101	E	109	M	117	U	125	}
86	V	94	^	102	F	110	N	118	V	126	-
87	W	95	—	103	G	111	O	119	W	127	DEL

Colors

Color	*Code #*
Transparent	1
Black	2
Medium Green	3
Light Green	4
Dark Blue	5
Light Blue	6
Dark Red	7
Cyan	8
Medium Red	9
Light Red	10
Dark Yellow	11
Light Yellow	12
Dark Green	13
Magenta	14
Gray	15
White	16

You need to know the numbers for the colors when you want to change the color of the screen. For example, CALL SCREEN(2) will give you a black screen! You also need to know the colors to change the colors of the characters and to do graphics. For example, CALL COLOR (3,14,6) will make all characters in set 3 magenta on a light blue background. CALL COLOR (3,6,6) will give you plain blue squares.

Graphics Sheet

There are 24 rows and 32 columns on the screen. When printing messages, the computer uses only 28 printing positions in each line, boxes 3 through 30. For graphics, though, it utilizes all 32 columns.

To print or draw at a certain position on the screen, you have to do a lot of counting! PRINT TAB(3);"LOOK AT THIS!" counts across to the third position before printing your message.

CALL HCHAR and CALL VCHAR are used with graphics. To draw something at a certain position on the screen, you find the position by counting *rows* first, then *columns.* For example, the position 5,3 is down 5 rows and across 3 columns. Or, put another way, it is the place where row 5 meets column 3.

To do well with graphics, you have to be able to find positions on the screen very easily. So practice!

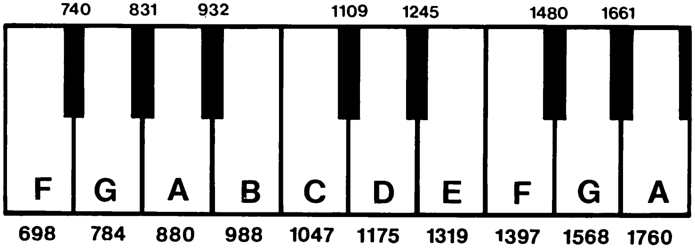
COLUMNS

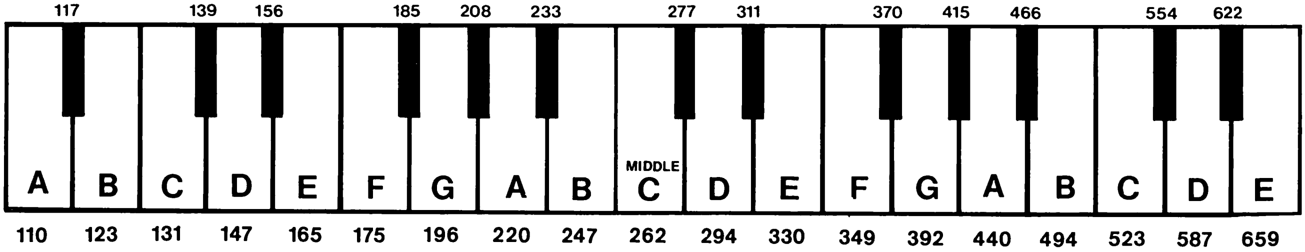
ROWS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1																																
2																																
3																																
4																																
5																																
6																																
7																																
8																																
9																																
10																																
11																																
12																																
13																																
14																																
15																																
16																																
17																																
18																																
19																																
20																																
21																																
22																																
23																																
24																																

Musical Notes

Use the numeric equivalents of the notes to create your own computer songs!





Notes

This page is for you to write on.

$ 6.95

CLAIRE BAILEY PASSANTINO

SCHOOL DAYS

for the TI-99/4A

School days, school days,

dear old Golden Rule days!

In spite of what some children say, school *is* fun! Claire Bailey Passantino has created bit-sized computer programs that allow your child to relive the exciting anticipation-filled first day of school, the often funny excuses for lost homework, challenging math and spelling quizzes . . . even the after-school basketball games.

School Days, another Itty Bitty Bytes book in the Creative Pastimes series, encourages your child to explore the magic of your home computer. The short, easy-to-do programming activities are designed to delight your youngster's eyes, ears and mind. Each program is accompanied by programming hints and suggestions for parents and teachers.

Help your child learn, grow and play—the computer way!

Some of the activities in School Days include:
Clothing the Bod ★ The Dog Ate It ★ Spelling Counts! ★ State the States ★ Jump Rope ★ Science ★ Report Card ★ The Big Game ★ Field Trip ★ Flashy Flashcards

Other Itty Bitty Bytes books:
Itty Bitty Bytes of Space
Matilda the Cat

For more information on Creative Pastimes Books
write to:

Reston Computer Group
Reston Publishing Company, Inc.
11480 Sunset Hills Road
Reston, Virginia 22090

0 21898 68875 0

0-8359-6887-1